Rapport Projet C/JAVA

# Introduction

Ce projet de programmation nous a été donné dans le cadre du cours de C / Java. L’objectif de ce cours est de mettre en pratique un certain nombre d’enseignements et d’approfondir certains points précis qui leurs sont liés.

Ainsi, le projet Chatrooms avait pour objectif principal de nous faire travailler avec un système de communication réseau en Java : les sockets et de mettre en pratique une architecture fondamentale du monde professionnel : le client / serveur.

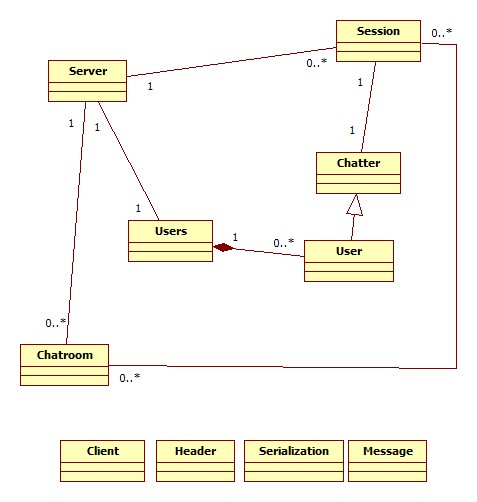
Afin de nous aider, et de nous mettre dans des conditions plus professionnelles que d’ordinaire, la partie d’analyse était déjà réalisée et une architecture était proposée. De plus, les fonctionnalités attendues, ainsi que l’aspect visuel final étaient également présentés dans le document du projet. Par conséquent, nous avons pu nous lancer rapidement dans le développement de la solution, tout en ajustant l’architecture selon nos besoins et notre vision du projet.

Dans ce rapport, nous allons vous décrire notre solution, nos choix de programmation, notre rendu final, ainsi que les erreurs commises et amélioration possibles.

Pour lancer l’application, lancez le main de la classe server/Server.java. Pour lancer une instance de l’application, lancez le main de la classe view/Application.java

## Choix de programmation

## Diagramme de classes



Nous avons légèrement modifié le diagramme de classe donné dans l’énoncé du TP. Dans notre modélisation, une « Chatroom » possède une liste de « Session » et une « Session est lié à un « Chatter ». Nous avons retiré la liste de « Chatter » dans la « Chatroom » pour éviter une redondance.

La conception quant à elle reste sensiblement la même. Le client et le serveur sont liés grâce à un socket et les entrées/sorties de ce socket sont utilisés dans un thread « Client » et un thread « Session ». Le client envoie des messages dans le socket. La session le reçois, fais les traitements avec l’aide du serveur et répond au client.

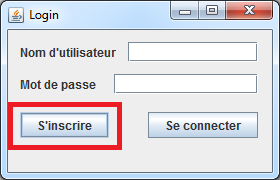
En l’état de notre application, il n’y a qu’un seul type d’utilisateur. Par conséquent, il est capable de tout faire : créer des chatrooms, supprimer des chatrooms, éteindre le serveur, etc.

## Fonctionnalités réalisées

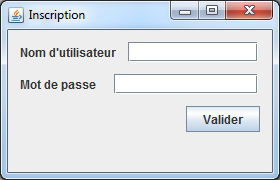
*Le serveur doit être lancé indépendamment des applications sur la machine locale. Il est possible de lancer plusieurs applications de chat sur une même machine. Cependant, dans le code de l’application, il est facile de remplacer l’adresse du serveur permettant ainsi de faire fonctionner le serveur sur une machine que localement (cela fonctionne d’après nos tests).*

### Inscription

Le premier pas à faire pour pouvoir se connecter au serveur est de s’inscrire. Pour cela, il faut utiliser le formulaire d’inscription de l’application.



En cliquant sur « S’inscrire »

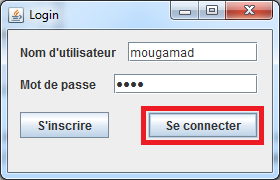


Une fois le formulaire validé, vous êtes redirigé vers la page de connexion où vous pouvez vous identifier avec votre couple pseudo / mot de passe renseigné pendant l’inscription.

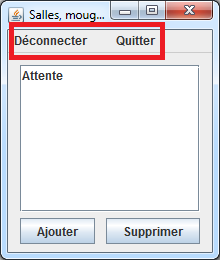
*N.B. : il n’est pas possible de s’inscrire avec un pseudo déjà utilisé sur le serveur. Cependant, aucun message d’information ne vous permettra de savoir si lors de votre inscription le pseudo existe déjà.*

### Connexion / Deconnexion

Une fois inscris, vous pouvez vous connecter depuis la page d’accueil de l’application. Pour cela vous devez entrer un couple pseudo / mot de passe connu du serveur et cliquer sur le bouton « connexion ».



Une fois connecté, vous arriverez sur la page présentant la liste des chatrooms du serveur. Une barre de menu propose deux boutons : « Déconnecter » et « Quitter ».



Pour vous déconnecter, il vous suffit de cliquer sur le menu « Déconnecter ». Vous serez alors redirigé vers la page de connexion de l’application.

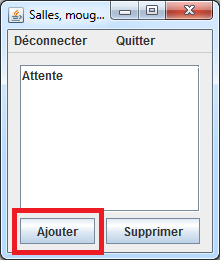
### Arrêter l’application

Pour arrêter l’application, vous avez deux possibilités :

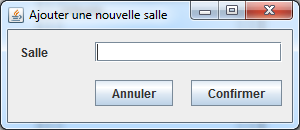
* Cliquer sur la croix en haut à droit de l’application
* Cliquer sur le bouton « Quitter » sur l’écran de sélection des chatrooms si vous êtes connecté (voir ci-dessus).

### Ajouter une chatroom

Pour pouvoir ajouter une chatroom, il faut être connecté et se trouver sur la page de sélection des chatrooms. Un bouton « Ajouter » vous est alors présenté en bas à gauche de la page.

**

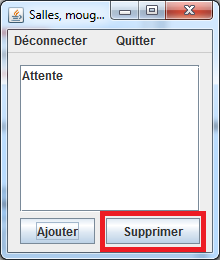
En cliquant sur ce bouton, vous arrivez sur un formulaire permettant de nommer la nouvelle chatroom qui sera créée.



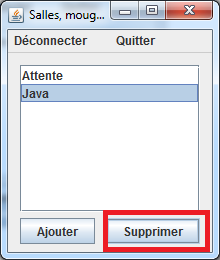
Il ne vous reste plus qu’à confirmer la création en cliquant sur le bouton « Confirmer ». Vous pouvez aussi choisir de ne pas créer la chatroom, cliquez alors sur le bouton « Annuler » et le formulaire se fermera.

### Suppression d’une chatroom

Pour supprimer une chatroom, il faut être connecté et se trouver sur la page de sélection des chatrooms. Un bouton « Supprimer » vous est alors présenté en bas à droit de la page.

**

Afin de supprimer une chatroom, il faut la sélectionner dans la liste des chatrooms et cliquer sur le bouton « Supprimer ».

**

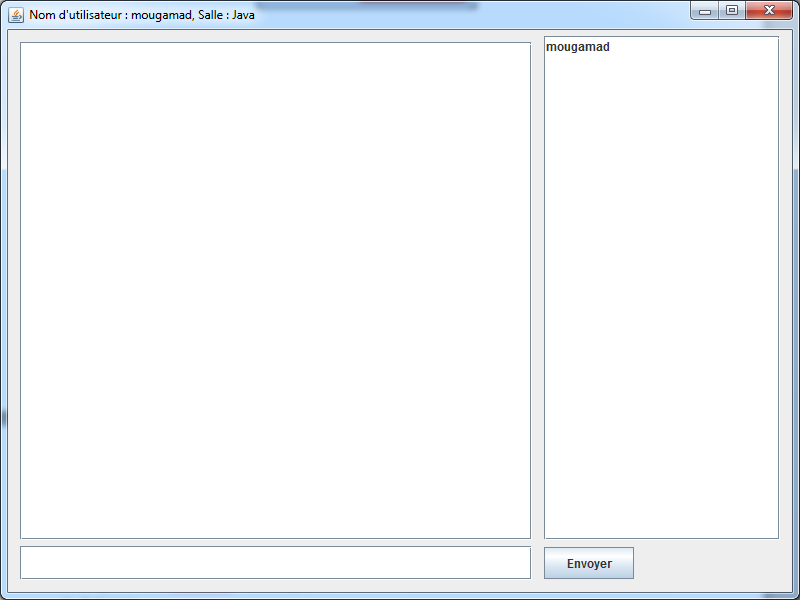
*N.B. : il n’est pas possible de supprimer la chatroom « Attente » qui est une chatroom instanciée au lancement du serveur.*

La chatroom est alors supprimée de la liste des chatrooms disponibles, du serveur, et l’ensemble des utilisateurs connectés dessus en sont donc déconnectés (leurs fenêtres de chat sur cette chatroom est fermée).

### Rejoindre une chatroom

Pour rejoindre une chatroom, il faut être connecté et se trouver sur la page de sélection des chatrooms. A partir de là, il vous suffit de double cliquer sur le nom de la chatroom que vous souhaitez rejoindre.

Une fois ce double clic effectué, la fenêtre vous permettant de communiquer avec les autres utilisateurs présents dans la chatroom s’ouvrira.



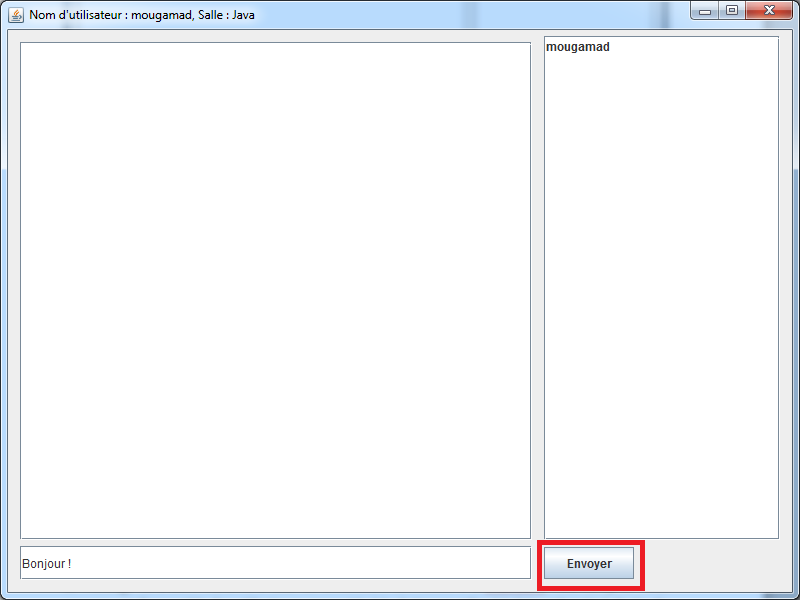
### Quitter une chatroom

Pour quitter une chatroom, il faut cliquer sur la croix en haut à droite de la fenêtre de discussion de la chatroom.

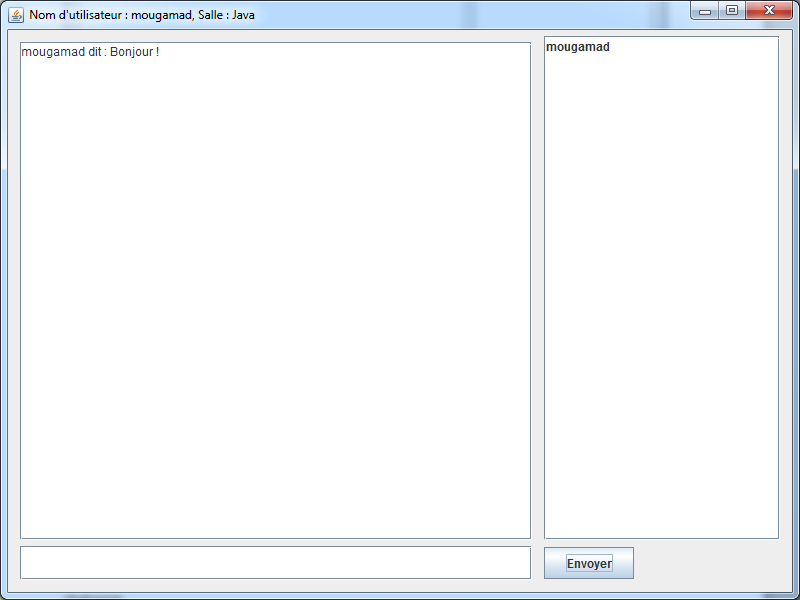
### Envoyer un message dans une chatroom

Pour envoyer un message dans une chatroom, il faut écrire ce dernier dans le champ en bas de la page de discussion de la chatroom.

Une fois le message rédigé, il vous reste deux possibilités pour l’envoyer à tous les utilisateurs de la chatroom : appuyer sur la toucher « Entrée » de votre clavier ou cliquer sur le bouton « Envoyer » prévu à gauche du champ.

**

Le message sera alors transmis à tous les utilisateurs connectés à la chatroom, vous compris. Vous pouvez voir l’ensemble des utilisateurs connectés à la chatroom grâce à la liste présente à droite de la page de discussion de la chatroom.



### Arrêter le serveur

Pour arrêter le serveur, il suffit d’écrire le message « /shutdown » dans une fenêtre de discussion de chatroom.

Le serveur s’arrêtera alors tout seul, sauvegardant la liste des utilisateurs enregistrés (sérialisation dans le fichier users.ser), ainsi que la liste des chatrooms ouvertes (sérialisation dans le fichier chatrooms.ser). Par conséquent, au prochain démarrage du serveur, ces données seront rechargées et le serveur sera dans le même état que lors de sa fermeture.

## Bugs connus

Il n’y a à ce jour aucun bug connu.

## Amélioration possibles

L’application fonctionne correctement et respecte les exigences du projet, cependant des améliorations sont envisageables :

* Donner des informations sur l’état de l’inscription (réussite, échec, pseudo existant, …)
* La création de plusieurs types d’utilisateurs avec des droits différents (seuls les admins pourraient éteindre le serveur par exemple)
* Créer des chatrooms avec mot de passe, empêchant ainsi à toute personne ne connaissant pas ce dernier de s’y connecter
* L’impossibilité de supprimer une chatroom non vide
* Mise en place d’un champ permettant de renseigner l’adresse IP à laquelle se trouve le serveur

# Conclusion

Ce projet a été l’occasion d’apprendre à gérer et à utiliser les sockets en Java. Ce sont les éléments qui nous ont le plus posé de difficultés car c’était la première fois que nous les utilisions et nombre de fonctions sont bloquantes dans les divers objets qui servent à exploiter correctement les communications au travers du socket.

La seconde difficulté a consisté à la mise en place de threads et leur bonne interruption au moment où les objets principaux étaient arrêtés également.

Il reste encore beaucoup à faire pour en faire un vrai projet de chat à part entière, mais en l’état notre solution est déjà exploitable.

Ce projet nous a vraiment apporté de nouvelles connaissances et fait partie des projets qui sont ludiques et qui font bien progresser, même des personnes connaissant Java.